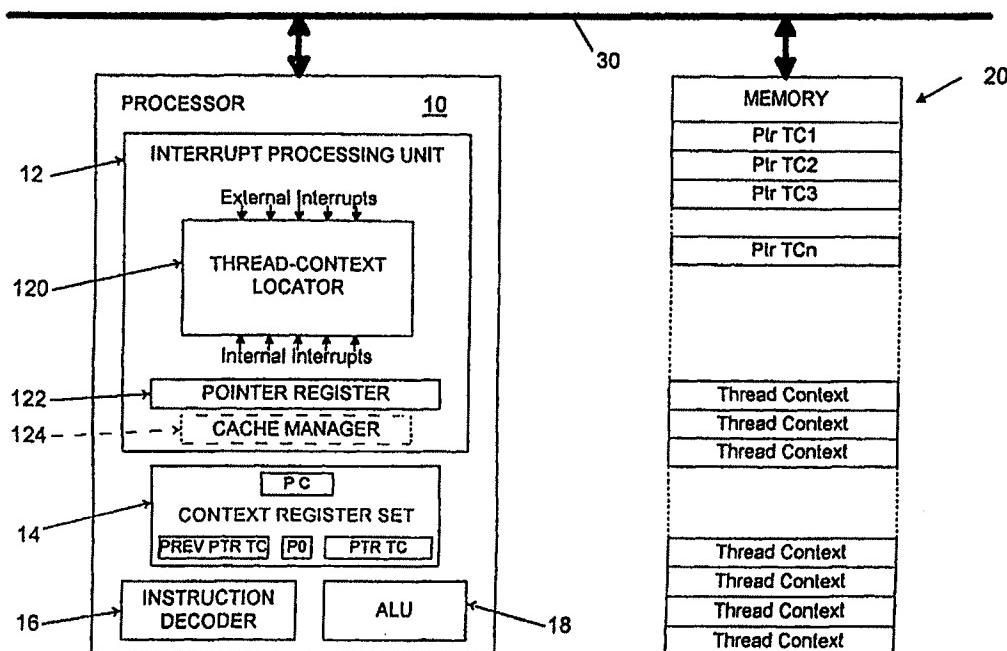




## INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 :	A3	(11) International Publication Number: WO 00/38060
G06F 9/46		(43) International Publication Date: 29 June 2000 (29.06.00)
(21) International Application Number: PCT/EP99/10170		(81) Designated States: CN, JP, KR, SG, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).
(22) International Filing Date: 16 December 1999 (16.12.99)		
(30) Priority Data:		Published <i>With international search report.</i>
09/218,551 22 December 1998 (22.12.98) US		(88) Date of publication of the international search report: 26 October 2000 (26.10.00)
09/273,938 22 March 1999 (22.03.99) US		
(71) Applicant: KONINKLIJKE PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).		
(72) Inventors: SAVILLE, Winthrop, L.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). ROSS, Kevin; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).		
(74) Agent: FAESSEN, Louis, Marie, Hubertus; Internationaal Octrooibureau B.V., Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).		

(54) Title: INTERRUPT/SOFTWARE-CONTROLLED THREAD PROCESSING



(57) Abstract

Rapid thread processing is achieved by transferring complete thread contexts between a memory and a context register set. Each thread context is read from a respective memory location in response to either a designated interrupt or an instruction.

(19)日本国特許庁 (JP)

(12) 公表特許公報 (A)

(11)特許出願公表番号

特表2002-533807

(P2002-533807A)

(43)公表日 平成14年10月8日(2002.10.8)

(51)Int.Cl.<sup>7</sup>

G 0 6 F 9/46

識別記号

3 1 1

3 1 3

3 4 0

F I

G 0 6 F 9/46

デーマコト<sup>\*</sup>(参考)

3 1 1 A 5 B 0 9 8

3 1 3 Z

3 4 0 B

審査請求 未請求 予備審査請求 未請求(全33頁)

(21)出願番号 特願2000-590052(P2000-590052)  
(86) (22)出願日 平成11年12月16日(1999.12.16)  
(85)翻訳文提出日 平成12年8月22日(2000.8.22)  
(86)国際出願番号 PCT/EP99/10170  
(87)国際公開番号 WO00/38060  
(87)国際公開日 平成12年6月29日(2000.6.29)  
(31)優先権主張番号 09/218, 551  
(32)優先日 平成10年12月22日(1998.12.22)  
(33)優先権主張国 米国(US)  
(31)優先権主張番号 09/273, 938  
(32)優先日 平成11年3月22日(1999.3.22)  
(33)優先権主張国 米国(US)

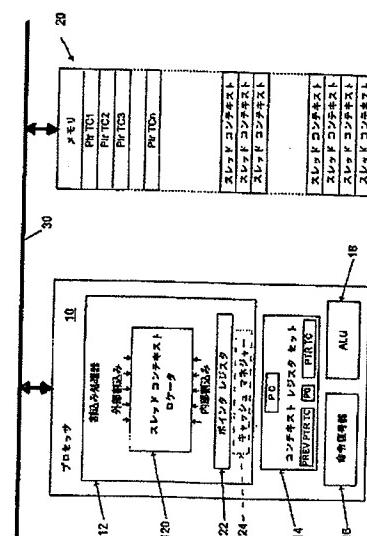
(71)出願人 コーニンクレッカ フィリップス エレクトロニクス エヌ ヴィ  
Koninklijke Philips Electronics N.V.  
オランダ国 5621 ベーー アイントーフェン フルーネヴァウツウェッハ 1  
(72)発明者 ウィンスロップ、エル. サビル  
オランダ国5656、アーアー、アインドーフェン、プロフ. ホルストラーン、6  
(72)発明者 ケビン、ロス  
オランダ国5656、アーアー、アインドーフェン、プロフ. ホルストラーン、6  
(74)代理人 弁理士 佐藤 一雄 (外3名)

最終頁に続く

(54)【発明の名称】 割込み/ソフトウェア制御スレッド処理

(57)【要約】

【解決手段】 メモリとコンテキスト・レジスタ・セットとの間で完全なスレッド・コンテキストを転送することによって迅速なスレッド処理が行われる。各スレッド・コンテキストは、指定された割込みまたは命令に応答してそれぞれのメモリ位置から読み出される。



**【特許請求の範囲】****【請求項1】**

コンテキスト・レジスタ・セットへのそれぞれのスレッド・コンテキストの読み込みに応答して、複数の異なるスレッドのうちのいずれかを選択的に実行するために単一のプロセッサを動作する方法において、

- a. 複数の割込みのそれに、それぞれのメモリ位置を関連付けることと、
- b. 複数のスレッド・コンテキスト・ポインタのぞれそれを、前記メモリ位置に保存し、このポインタのそれぞれは、前記スレッドのうちの1つのスレッドの動作状態を十分に記述するスレッド・コンテキストを含むためのメモリ位置を特定することと、
- c. いずれかの割込みの発生に応答して、前記割込みに関連付けられているそれぞれのメモリ位置からポインタを読み出すこと、
- d. スレッド・コンテキスト・ポインタ読み出しによって特定されるメモリ位置から、スレッド・コンテキストをコンテキスト・レジスタ・セットに読み込むこと、
- e. スレッド・コンテキスト読み出しによって記述されているスレッドを実行すること、

を備える単一のプロセッサを動作する方法。

**【請求項2】**

請求項1記載の方法であって、スレッド・コンテキスト・ポインタのそれぞれが、異なるメモリ位置を特定する方法。

**【請求項3】**

請求項1記載の方法であって、1より多いスレッド・コンテキスト・ポインタが、同じメモリ位置を特定する方法。

**【請求項4】**

請求項1記載の方法であって、プロセッサにより実行されているプログラム・ストリーム中における割込み命令の1つ又は複数のスレッド変更命令の1つに応答して実行しているスレッドを、プロセッサが変更する方法。

**【請求項5】**

請求項4記載の方法であって、スレッド変更命令は、現在実行しているスレッドのコンテキストを保存するため、及び、実行されているスレッドを異なるスレッドに変更するための、SWITCH命令を含む方法。

#### 【請求項6】

請求項5記載の方法であって、

プロセッサは、先に実行したスレッドに対するコンテキストのアドレスを保存し、

前記スレッド変更命令は、実行しているスレッドを、前記先に実行したスレッドに変更するための第1のSWITCH命令を含む方法。

#### 【請求項7】

請求項5記載の方法であって、第2のSWITCH命令を含み、この第2のSWITCH命令は、第2のSWITCH命令により特定されているメモリ位置に含まれているそのコンテキストを有するスレッドに、実行されているスレッドを変更するためのものである方法。

#### 【請求項8】

請求項7記載の方法であって、第2のSWITCH命令は、前記コンテキストのメモリ位置を暗黙的に特定する方法。

#### 【請求項9】

請求項7記載の方法であって、第2のSWITCH命令は、前記コンテキストのメモリ位置を明示的に特定する方法。

#### 【請求項10】

請求項9記載の方法であって、第2のSWITCH命令は、前記コンテキストのメモリ位置を特定するポインタを含む方法。

#### 【請求項11】

請求項9記載の方法であって、第2のSWITCH命令は、前記コンテキストのメモリ位置を特定するポインタを含むメモリ位置を特定する方法。

#### 【請求項12】

請求項4記載の方法であって、スレッド変更命令は、現在実行しているスレッドのコンテキストを保存することなく、異なるスレッドを実行するためのRES

T O R E 命令を含む方法。

【請求項13】

請求項12記載の方法であって、

プロセッサは先に実行したスレッドに対するコンテキストのアドレスを保存し

前記スレッド変更命令は、実行されているスレッドを前記先に実行したスレッドに変更するための第1のR E S T O R E 命令を含む方法。

【請求項14】

請求項12記載の方法であって、第2のR E S T O R E 命令を含み、この第2のR E S T O R E 命令は、第2のR E S T O R E 命令により特定されているメモリ位置に含まれているそのコンテキストを有するスレッドに、実行されているスレッドを、変更するためのものである方法。

【請求項15】

請求項14記載の方法であって、第2のR E S T O R E 命令は前記コンテキストのメモリ位置を暗黙的に特定する方法。

【請求項16】

請求項14記載の方法であって、第2のR E S T O R E 命令は前記コンテキストのメモリ位置を明示的に特定する方法。

【請求項17】

請求項16記載の方法であって、第2のR E S T O R E 命令は前記コンテキストのメモリ位置を特定するポインタを含む方法。

【請求項18】

請求項16記載の方法であって、第2のR E S T O R E 命令は前記コンテキストのメモリ位置を特定するポインタを含むメモリ位置を特定する方法。

【請求項19】

請求項4記載の方法であって、スレッド変更命令は、割込みをエミュレートするためのS/W I N T E R R U P T 命令を含む方法。

【請求項20】

請求項4記載の方法であって、複数のコンテキスト・レジスタ・セットが設け

られ、前記单一のプロセッサは前記コンテキスト・レジスタ・セットの所定の1つに含まれている方法。

#### 【請求項21】

請求項1記載の方法であって、コンテキスト・レジスタ・セットは、

- a. プロセッサにより現在実行されているスレッドのコンテキストのメモリ・アドレスを保存するためのレジスタと、
  - b. スレッド・コンテキスト・ポインタによって特定されていないスレッド・コンテキストの位置を特定するために用いられるレジスタと、
- を含む方法。

#### 【請求項22】

請求項1記載の方法であって、コンテキスト・レジスタ・セットはプロセッサで先に実行されたスレッドのコンテキストのメモリ・アドレスを保持するためのレジスタを含む方法。

#### 【請求項23】

請求項1記載の方法であって、プロセッサは、複数のコンテキスト・レジスタ・セットを含む方法。

#### 【請求項24】

請求項23記載の方法であって、コンテキスト・レジスタ・セットの数は、プロセッサで実行すべきスレッドの数より、少ない方法。

#### 【請求項25】

請求項24記載の方法であって、スレッド・コンテキストが読み込まれるコンテキスト・レジスタ・セットを選択するために、優先順位を基にしたアルゴリズムが採用される方法。

#### 【請求項26】

請求項24記載の方法であって、

スレッドはより高い優先順位のスレッド及びより低い優先順位のスレッドを含み、

より低い優先順位のスレッドよりも、より高い優先順位のスレッドが利用可能なように、比例して大きい数のコンテキスト・レジスタ・セットが形成される、

される方法。

#### 【請求項27】

請求項26記載の方法であって、少なくとも1つのスレッドの優先順位を、プロセッサの動作中に変更できる方法。

#### 【請求項28】

請求項26記載の方法であって、

より高い優先順位のスレッド又はより低い優先順位のスレッドの1つのためのスレッド・コンテキストの読み出し時に、セット選択アルゴリズムが用いられ、

それぞれの優先順位で使用可能に形成されたコンテキスト・レジスタ・セットのすべてが、読み出されているもの以外のスレッドのコンテキストを既に含んでいる方法。

#### 【請求項29】

請求項28記載の方法であって、前記アルゴリズムは、前記スレッド・コンテキストを、最後に保存されたスレッド・コンテキストを含んでいる使用可能なコンテキスト・レジスタに、読み込む方法。

#### 【請求項30】

コンテキスト・レジスタ・セットへのそれぞれのスレッド・コンテキストの読み込みに応答して、複数の異なるスレッドのうちのいずれかを、複数のプロセッサのぞれぞれが選択的に実行するようにプロセッサのネットワークを動作する方法において、

- a. 複数の割込みのそれに、それぞれのメモリ位置を関連付けることと、
- b. 複数のスレッド・コンテキスト・ポインタのそれぞれを、前記メモリ位置に保存し、このポインタのそれぞれは、前記スレッドのうちの1つのスレッドの動作状態を十分に記述するスレッド・コンテキストを含むためのメモリ位置を特定することと、
- c. いずれかの割込みの発生に応答して、前記割込みに関連させられているそれぞれのメモリ位置からポインタを読み出すことと、
- d. スレッド・コンテキスト・ポインタ読み出しによって特定されるメモリ位置から、スレッド・コンテキストをコンテキスト・レジスタ・セットに読み込むこと

と、

e. スレッド・コンテキスト読出しによって記述されているスレッドを実行することと、

を備えるプロセッサのネットワークを動作する方法。

【請求項31】

請求項30記載の方法であって、スレッドの少なくとも1つが、1より多いプロセッサにより実行できる共通スレッドである方法。

## 【発明の詳細な説明】

### 【0001】

#### 発明の背景

##### 1. 発明の分野

本発明はスレッド指向処理 (thread-oriented processing) に関するものであり、更に詳しくいえば、コンテキストの使用による多重スレッド処理 (multiple-thread) に関するものである。

### 【0002】

##### 2. 関連技術についての説明

多重スレッド処理においては、プロセッサは複数の異なる処理（一般に「スレッド」と呼ばれている）を逐次実行できる。所定の時間期間の経過後または特定の命令の受け取り時などの、特定の事象の発生時に、プロセッサは1つのスレッドの実行を保留し、保留するスレッドの状態を記述しているコンテキストを保存し、別のスレッドの実行を開始する。そのスレッドもそれぞれのコンテキストにより記述されている。各「コンテキスト」は、プロセッサが新しいスレッドの実行を開始するため、または保留されているスレッドの動作を継続するために必要な情報を含んでいる。通常はこの情報はメモリ・アドレスと、ステータス情報と、データとを含んでいる。

### 【0003】

保存されているスレッド・コンテキストの迅速なアクセスを直接行うために、いくつかのプロセッサはローカル・コンテキスト・レジスタの多数のバンクを含んでいる。しかし、これは硬直した構成であって、実行できるスレッドの数を一定数に制限する。また、設けられているレジスタ・バンクの数よりスレッドの数が少ない時は常に効率が悪い。

### 【0004】

あるいは、多数のコンテキストを別々のメモリに保存できる。そのようなやり方の種々の例が米国特許第5349680号に記述されている。その特許に従来の情報処理装置として記述されている、それらの例の1つにおいては、主プロセッサは、種々のアプリケーション・プロセスから動作を逐次実行するためのアプ

リケーション・サポート装置と、情報処理装置における動作を制御するためのシステム・サポート装置とを含んでいる。この装置の効率は低いと記述されている。その理由は、アプリケーション・サポート装置とシステム・サポート装置は同時に決して動作させられないからである。米国特許第5349680号は保存されている多数のコンテキストを利用する別の装置を提案しているが、それらの装置のおおのは2つの別々のプロセッサを使用することを要する。これは効率を向上するための費用のかかるやり方である。

### 【0005】

#### 発明の概要

本発明の目的は、一定数のローカル・コンテキスト・レジスタを使用するという硬直性の不利益をこうむらない、単一プロセッサによる迅速かつ効率的な多重スレッド処理を可能にすることである。

### 【0006】

この目的およびその他の目的は、コンテキストをコンテキスト・レジスタ・セットとメモリとの間で交換（swapping）することによって達成される。これによって、最少のハードウェアにより多重スレッド処理が可能にされる。本発明によれば、

- ・複数の割込みのそれぞれに、それぞれのメモリ位置を関連付けること、
- ・複数のスレッド・コンテキスト・ポインタのそれぞれを、前記メモリ位置に保存し、このポインタのそれぞれは、前記スレッドのうちの1つのスレッドの動作状態を十分に記述するスレッド・コンテキストを含むためのメモリ位置を特定すること、
- ・いずれかの割込みの発生に応答して、前記割込みに関連させられているそれぞれのメモリ位置からポインタを読出すこと、
- ・スレッド・コンテキスト・ポインタ読出しによって特定されているメモリ位置から、スレッド・コンテキストをコンテキスト・レジスタ・セットに読込むこと、
- ・スレッド・コンテキスト読出しによって記述されているスレッドを実行すること、

を含む方法が用いられる。

#### 【0007】

それぞれの割込みに関連させられているメモリ位置に、コンテキスト自体ではなくて、コンテキストのポインタを保存することによって、割込みは特定のコンテキストから関係を断たれる。これによって割込みに対する応答の決定における融通性が高くなる。また、ただ1つのメモリ位置に1より多い割込みに共通のコンテキストを保存し、且つ、この共通コンテキストのアドレスをそれらの割込みのための各ポインタに含めることによって、メモリを節約できる。更に、コンテキスト自体をポインタにより特定されているメモリ位置に直接保存することによって、1つのスレッドから別のスレッドへの迅速なプロセッサ変更が可能にされる。

#### 【0008】

ここで使用している「メモリ」という語は、それが用いられる態様に一致するものと一般に解釈されるべきであることを意図しており、かつ、制約なしに、レジスタ、RAM、DRAM、ROM、及び、それらの装置の組合せなどの、各種の揮発性装置および不揮発性装置を含むことに注目されたい。また、「読み出し」は、1つのメモリからの情報の検索、及び、他のメモリへのその書き込みを意味する。

#### 【0009】

本発明の特に有利な実施例では、プロセッサによって実行されているプログラム・ストリーム（すなわち、そのプログラムを構成している命令列）中のある命令が、プロセッサにおいてコンテキストの変更を直接行う。これによってスレッド自体による迅速なコンテキスト変更の開始が可能になる。

#### 【0010】

##### 【発明の実施の形態】

##### 好適な実施例についての説明

図1の多重スレッド処理装置1は、単一のプロセッサ10とメモリ20を含んでいる。例示したプロセッサ10は、プログラム・カウンタ・レジスタにより特定された命令を順次実行するためにクロックパルスを利用する、ハードウェアで

加速された装置 (hardware-accelerated unit) である。通常は、プログラム・カウンタ・レジスタは、プロセッサが読出して、実行する次の命令のメモリ位置を含んでいる。

#### 【0011】

プロセッサは、割込み処理器 12 と、コンテキスト・レジスタ・セット 14 と、命令復号器 16 と、算術論理装置 18 とを含む。この実施例では、メモリ 20 は、とくに、複数のスレッド・コンテキスト・ポインタ  $\text{P}_{\text{tr}} \text{TC}_1, \text{P}_{\text{tr}} \text{TC}_2, \text{P}_{\text{tr}} \text{TC}_3, \dots, \text{P}_{\text{tr}} \text{TC}_n$  と、複数のスレッド・コンテキストを保存するための多数のメモリ位置を有する RAM を備えている。

#### 【0012】

プロセッサ 10 とメモリ 20 は、共通バス 30 に接続されて相互に通信するとともに、このバスに接続されている他のハードウェアとも通信する。このバスは、アドレス、割込み、データ、読み出しきリストローブ、書き込みストローブ、装置選択ストローブなどの情報を伝えるためのそれぞれのラインを含んでいる。好ましくは、このバスは、プロセッサおよびメモリと共通のシリコン基板上に少なくとも部分的に形成された高速バスである。

#### 【0013】

プロセッサ 10 の動作は割込みおよびプログラム・ストリーム中の命令によって全体的 (entirely) に制御される。割込みは、バス 30 から受けられた外部割込み、またはプロセッサ自体で、たとえば、プロセッサ内のタイマ (図示せず) から、発生された内部割込みとすることができます。各外部割込みには、メモリ 20 内の、スレッド・コンテキスト・ポインタ  $\text{P}_{\text{tr}} \text{TC}_1, \text{P}_{\text{tr}} \text{TC}_2, \text{P}_{\text{tr}} \text{TC}_3, \dots, \text{P}_{\text{tr}} \text{TC}_n$  の 1 つが保存されている所定の位置が関連付けられている。それらのポインタのおののには、バス 30 に接続されているハードウェアにより利用されるスレッド・コンテキストの 1 つが関連付けられ、特定のコンテキストが保存されているメモリ位置を特定する。1 より多いポインタ (たとえば、 $\text{P}_{\text{tr}} \text{TC}_1$  および  $\text{P}_{\text{tr}} \text{TC}_3$ ) に同じスレッド・コンテキストを関連付けることができ (すなわち、「共用する」) 、したがって、同じメモリ位置を特定する。これは、たとえば、バス 30 に接続されている 1 より

多い装置、たとえば、受信FIFOおよび送信FIFO、が同じコンテキストを利用するならば、有用である。内部割込みのそれぞれが、メモリ20内の、プロセッサにより利用されているスレッド・コンテキストの1つが保存されている所定の位置に関連付けられる。

#### 【0014】

あるコンテキストが共用されているならば、処理時間が長くなる。その理由は、割込みサービス・スレッドが割込みの元を決定しなければならないからである。しかし、より少ないメモリが必要とされる。逆に、ある割込みがあるコンテキストを独占的に使用するものとすると、割込みサービス・スレッドは割込みの元を固有的に特定でき、処理応答時間はより短い。これによって装置のアーキテクチャに融通性が与えられる。

#### 【0015】

全ての割込みは、割込み処理器12によって処理される。この割込み処理器はスレッド・コンテキスト・ロケータ120とポインタ・レジスタ122を含み、かつキャッシュ・マネジャー124を含むことを選択できる。スレッド・コンテキスト・ロケータ120は、外部割込みおよび内部割込みに関連付けられているメモリ位置を識別するアドレスを生成する。任意の割込みに応答して、スレッド・コンテキスト・ロケータは、スレッド・コンテキスト・ポインタPtr\_TC1、Ptr\_TC2、Ptr\_TC3、... Ptr\_TCrの関連付けられている1つが保存されているメモリ20の所定の位置を特定するアドレスを生成する。その後で、プロセッサは、スレッド・コンテキスト位置をこのメモリ位置からポインタ・レジスタ122に読み込む。外部割込みの例には、FIFO TR ANSMIT COMPLETE、FIFO DATA RECEIVED、DMA TRANSFER COMPLETE、FIFO TRANSMIT FAILTURE等が含まれる。内部割込みの例には、内部タイマ等が含まれる。

#### 【0016】

好ましくは、スレッド・コンテキスト・ロケータは、プログラム可能なルックアップ・テーブルまたは符号器などのアドレスを発生するための専用の優先順位感知ハードウェアを備えており、両者ともこの技術で周知である。両者ともアド

レッシングの速さを最速化し、かつ、より低い優先順位の割込みより先により高い優先順位の割込みの処理を可能にする。

#### 【0017】

コンテキスト・レジスタ・セット14は、現在プロセッサにより実行されているスレッドのコンテキストを含む複数のレジスタを備えている。好適な実施例では、レジスタ・セット14は、以下を含んでいる。

#### 【0018】

- 現在プロセッサにより実行されているスレッドのためのコンテキストのメモリ・アドレスを保存するためのレジスタPTR\_TC。

#### 【0019】

- 先に (previously) プロセッサにより実行されたスレッドのためのコンテキストのメモリ・アドレスを保存するためのレジスタPREV\_PTR\_TC。

#### 【0020】

- メモリ20に保存されているスレッド・コンテキスト・ポインタにより特定されないことがあるスレッド・コンテキストの位置を特定するために用いられるレジスタP0。

#### 【0021】

- メモリ20内で次にアクセスすべき命令のアドレスを特定するために、継続的に更新するプログラム・カウンタ・レジスタPC。

#### 【0022】

- 1つまたは複数の汎用レジスタ (図示せず)。

#### 【0023】

- メモリ20から読み出され、又は、算術論理装置18で発生するデータを含むための1つまたは複数のデータ・レジスタ (図示せず)。

#### 【0024】

命令復号器16は、メモリ20から読み出された命令を、算術論理装置18によって処理すべきより低いレベルのオペレーション・コードに変換するため、シーケンサまたはマイクロシーケンサなどの、従来のハードウェア部品である。算術論理装置も従来のハードウェア部品である。

## 【0025】

図2は、順次発生する、複数の割込み例とコンテキスト変更のための複数の命令の例との制御の下にあるプロセッサ10の動作を示す。ボックス40はそれらの割込みおよびコンテキスト変更命令が起きるにつれてのプロセッサ内の事象列を表す。表されているコンテキスト変更命令の例(RESTORE PREV TC, SWITCH P0 TO TC, RESTORE FM P0 TC, SWITCH TO PREV TC, 及び、S/W INTERRUPT)がコンテキスト・レジスタ・セット14における変更を行う。これについては後で詳しく説明する。

## 【0026】

ボックス20'は、特定のポインタと、それらのポインタによって位置付けされたスレッド・コンテキストと、シーケンス40において発生する命令SWITCH TO P0 TC及びRESTORE FM P0 TCにより位置決めされたスレッド・コンテキストとを、含んでいる、メモリ20内の位置を表している。この特定の例では、メモリ位置20'は、以下を含んでいる。

## 【0027】

- 割込みINT 1に応答してプロセッサにより読み出され、かつ関連させられているスレッド・コンテキストTHREAD CONTEXT Fのメモリ位置を含むポインタPtr TC INT 1。

## 【0028】

- 割込みEXT 1に応答してプロセッサにより読み出され、かつ関連させられているスレッド・コンテキストTHREAD CONTEXT Aのメモリ位置を含むポインタPtr TC EXT 1。

## 【0029】

- 割込みEXT 2に応答してプロセッサにより読み出され、かつ関連させられているスレッド・コンテキストTHREAD CONTEXT Bのメモリ位置を含むポインタPtr TC EXT 2。

## 【0030】

- 割込みEXT 3に応答してプロセッサにより読み出され、かつ関連させられ

ているスレッド・コンテキストTHREAD CONTEXT Cのメモリ位置を含むポインタP t r T C E X T 3。

【0031】

- ・ スレッド・コンテキストTHREAD CONTEXT A。

【0032】

- ・ スレッド・コンテキストTHREAD CONTEXT B。

【0033】

- ・ スレッド・コンテキストTHREAD CONTEXT C。

【0034】

- ・ スレッド・コンテキストTHREAD CONTEXT D。

【0035】

- ・ スレッド・コンテキストTHREAD CONTEXT E。

【0036】

- ・ スレッド・コンテキストTHREAD CONTEXT F。

【0037】

この実施例では、命令RESTORE PREV TC、SWITCH TO P0 TC、RESTORE FM P0 TC、SWITCH TO PR EV TCおよびS/W INTERRUPTは、メモリ20に含まれており、それらのメモリ位置がプログラム・カウンタ・レジスタPCにより特定されるたびに、これらの命令が命令復号器16に読み込まれる。

【0038】

図2に示されているシーケンスは時刻t0で始まる。その時にはプロセッサ10は、コンテキスト・レジスタ・セットに含まれているTHREAD CONTEXT Aにより表されているスレッドAを実行している。簡単にするために、各割込みは可能にされていて、それが発生されると直ちに実行されると仮定することにする。

【0039】

時刻t1では、プロセッサ10は、バス30内の割込みラインの1つを介して外部割込みEXT 2を受ける。この割込みに応答して、プロセッサは、以下の

処理を行う。

【0040】

- コンテキスト・レジスタ・セット14に現在含まれているスレッド・コンテキスト、すなわち、スレッドAに対するコンテキスト(THREAD CONTEXT A)を、コンテキスト・レジスタ・セット14内のPTR\_TCレジスタに含まれている現在のスレッド・コンテキスト・アドレスにより特定されているメモリ位置に読み込む。

【0041】

- レジスタPTR\_TCに現在含まれているアドレスをレジスタPREV\_PTR\_TCに読み込む。

【0042】

- 外部割込みEXT 2に関連付けられているメモリ位置、すなわち、ポインタPTR\_TC EXT 2が保存されているメモリ位置、を特定するアドレスを、スレッド・コンテキスト・ロケータ120に発生させる。

【0043】

- THREAD CONTEXT Bのメモリ位置を特定しているポインタPTR\_TC EXT 2を、ポインタ・レジスタ122に読み込む。

【0044】

- このスレッド・コンテキストをコンテキスト・レジスタ・セット14に読み込み、スレッドBの実行を開始する。

【0045】

時刻t2では、プロセッサ10は、スレッドB内の最後の命令として、命令RESTORE PREV\_TCに出くわす。この命令の実行において、プロセッサは次の処理を実行する。

【0046】

- PREV\_PTR\_TCレジスタからのアドレスをポインタ・レジスタ122に読み込む。

【0047】

- 所望により、PTR\_TCレジスタからのアドレスをPREV\_PTR\_T

Cレジスタに読み込む。

#### 【0048】

- ・ ポインタ・レジスタ122によって特定されたメモリ20'位置からのスレッド・コンテキストAを、コンテキスト・レジスタ・セット14に読み込み、スレッドAの実行を再び開始する。

#### 【0049】

時刻t3では、スレッドAを実行している間に、プロセッサは、プログラム・カウンタ・レジスタPCにより特定されたメモリ20'内のそれぞれの位置から命令SWITCH TO P0 TCを読出す。この命令は1つのスレッドから、メモリ20'に予め保存されているスレッド・コンテキスト・ポインタPtr TC EXT1、Ptr TC EXT2または、Ptr TC EXT3のいずれによても特定されないことがある別のスレッドへの切換えを可能にするために与えられる。この切換えを容易にするために、SWITCH TO P0 TC命令に先行する命令の1つが、切換えるべきスレッドのためのコンテキストの位置を特定するアドレスを、コンテキスト・レジスタ・セット内のP0レジスタに予め保持する。この例では、切換えは、現在のスレッドAからメモリ20'内のどこかに配置されているスレッドDへである。この命令の実行においては、プロセッサは、以下の処理を実行する。

#### 【0050】

- ・ THREAD CONTEXT Aを、コンテキスト・レジスタ・セット14内のレジスタPTR TCに含まれている現在のスレッド・コンテキスト・アドレスによって特定されているメモリ位置に読み込む。

#### 【0051】

- ・ レジスタPTR TCに現在含まれているアドレスをレジスタPREV PTR TCに読み込む。

#### 【0052】

- ・ P0レジスタからのアドレスをポインタ・レジスタ122に読み込む、
- ・ ポインタ・レジスタ122により特定されているメモリ20'内の位置からのスレッド・コンテキスト(THREAD CONTEXT D)を、コンテキ

スト・レジスタ・セット14に読み込む。

【0053】

- スレッドDの実行を開始する。

【0054】

時刻t4では、スレッドDの実行中にメモリ20'から読出された命令に応答して、プロセッサ10は命令RESTORE FM P0 TCをメモリ20'内のそれぞれの位置から読出すことによってその命令を実行する。命令RESTORE FM P0 TCは、命令SWITCH TO P0 TCに類似するが、コンテキスト・レジスタ・セット14に現在のコンテキストを保存することなくスレッドのコンテキストを読出す。この命令の実行を容易にするために、RESTORE FM P0 TC命令に先行する命令の1つが、復帰すべきスレッドに対するコンテキストの位置を特定するアドレスをP0レジスタに予め保存する。この例では、プロセッサは、以下の処理によりスレッドEを復帰する。

【0055】

- P0レジスタからのTHREAD CONTEXT Eに対するアドレスをポインタ・レジスタに読み込む。

【0056】

- 所望により、PTR TCレジスタからのアドレスを、PREV PTR TCレジスタに読み込む。

【0057】

- メモリ20'からのTHREAD CONTEXT Eを、コンテキスト・レジスタ・セットに読み込む。

【0058】

時刻t5では、スレッドEの実行中に、プロセッサは、プログラム・カウンタ・レジスタPCにより特定されているメモリ20'内のそれぞれの位置から命令SWITCH TO P0 TCを読み出す。この命令は、現在実行中のスレッドから、先に実行されていたスレッドへの切換えを可能にするために与えられる。この例では、切換えは現在のスレッドEからスレッドDへである。この命令の実行に際しては、プロセッサは、以下の処理を行う。

## 【0059】

- THREAD CONTEXT Eを、コンテキスト・レジスタ・セット14内のレジスタ PTR TCに含まれている現在のスレッド・コンテキスト・アドレスによって特定されるメモリ20'内のメモリ位置に読み込む。

## 【0060】

- PREV PTR TCレジスタからのアドレスを、ポインタレジスタ122に読み込む。

## 【0061】

- レジスタ PTR TC内に含まれている現在のアドレスを、レジスタ PRE V PTR TCに読み込む、
- ポインタ・レジスタ122により特定されているメモリ20'の位置からのスレッド・コンテキストFを、コンテキスト・レジスタ・セット14に読み込み、スレッドDの実行を再び開始する。

## 【0062】

時刻t6では、スレッドDの実行中に、プロセッサは、プログラム・カウンタ・レジスタPCにより特定されているメモリ20'内のそれぞれの位置から命令S/W INTERRUPTを読出す。この命令は、ハードウェア割込みのソフトウェア・エミュレーションを可能にするために与えられる。この例では、その命令は、割込みINT 1をエミュレートする。この命令に応答して、プロセッサは、ハードウェア割込みINT 1を受けた時と同じように応答する。すなわち、プロセッサは、以下の処理を行う。

## 【0063】

- コンテキスト・レジスタ・セット14からの現在実行中のTHREAD CONTEXT Dを、レジスタ PTR TCに含まれているアドレスによって特定されているメモリ位置に読み込む。

## 【0064】

- レジスタ PTR TCに現在含まれているアドレスを、レジスタ PRE V PTR TCに読み込む。

## 【0065】

- ・ 割込みINT 1に関連付けられているメモリ位置、すなわち、ポインタPtr\_TC\_INT\_1が保存されているメモリ位置、を特定するアドレスをスレッド・コンテキスト・ロケータ120に発生させる。

#### 【0066】

- ・ THREAD\_CONTEXT\_Fのメモリ位置を特定している、ポインタPtr\_TC\_INT\_1を、ポインタ・レジスタ122に読み込む。

#### 【0067】

- ・ このスレッド・コンテキストを、コンテキスト・レジスタ・セット14に読み込み、スレッドFの実行を開始する。

#### 【0068】

これまで説明してきたように、プロセッサは単一のコンテキスト・レジスタ・セット14に含まれているコンテキストを迅速に変更する性能を有する。しかし、コンテキストを変更する速さを増すために、プロセッサは複数のコンテキスト・レジスタ・セットを有することが好ましい。その場合には、コンテキスト・キャッシュ・マネジャー124をプロセッサに含めることを選択できる。

#### 【0069】

最速化を達成するために、プロセッサにより実行されるべき種々のスレッドの数と同数のコンテキスト・レジスタ・セットが、キャッシュにある。しかし、そのような高速があらゆるスレッドに対して不要である場合には、キャッシュ・メモリ空間の使用が非効率的である。本発明の他の特徴によれば、スレッドより少ない数のコンテキスト・レジスタ・セットを有する。しかし、優先順位がより低いスレッドよりも、優先順位がより高いスレッドの方に、スレッド当たりに、より多くのレジスタ・セットを使用できるようにすることによって、最適な効率が達成される。図3および図4はそのような優先順位付けの2つの例を示す。

#### 【0070】

図3は、キャッシュメモリ内に配置されて、キャッシュ・マネジャー124の制御の下にある4つのコンテキスト・レジスタ・セット(I、II、III、IV)を用いることによって、8つのスレッド(スレッドA、B、... H)を実行するように構成されている装置におけるスレッド処理を表す。各コンテキスト

・レジスタ・セットは図1に示されているそれに類似するが、8つのスレッドのおののおののコンテキストに含まれている優先順位コード $p$ を保持するレジスタを更に含む。この例では、

- ・高い優先順位 ( $p = 1$ ) のスレッドまたは低い優先順位 ( $p = 0$ ) のスレッドを指示するために単一のビットが用いられる、
- ・スレッドA、B、Cは高い優先順位のスレッドとして指定されている、
- ・スレッドD、E、F、G、Hは低い優先順位のスレッドとして指定されている。

#### 【0071】

図3の例では、時刻 $t_0$ でプロセッサ10は、どのスレッドも実行しておらず、コンテキスト・レジスタ・セットI、II、III、IVは、それらのレジスタの1つもスレッド・コンテキストを含んでいない状態であるリセット状態にある。プロセッサはそれが次のようにして割込まれるまでアイドル状態にある。

#### 【0072】

- ・時刻 $t_1$ では、割込みEXT2がバス30を介して受けられる。プロセッサは(スレッド・コンテキスト・ロケータ120を介して)ポインタ $P_{tr\ T} C_{TC\ EXT\ 2}$ を、ポインタ・レジスタ122に読み込み、その後でTHREAD CONTEXT Bをこのポインタにより特定されている、メモリ位置から読み出す。キャッシュ・マネジャー124は、このスレッド・コンテキストが高い優先順位のコード $p = 1$ を含むことと、全部で4つのコンテキスト・レジスタ・セットがフリーである(すなわち、どのコンテキストも含んでいない)こと、を判定し、THREAD CONTEXT Bをコンテキスト・レジスタ・セットIへ送る。(この例では、キャッシュ・マネジャーは最初のフリーなレジスタ・セットを数値的に常に選択するが、これは、ランダムに含んでいる、任意のシーケンスにおいて行うことができる。)コンテキスト・レジスタ・セットIは今はアクティブとなり、プロセッサがスレッドBを実行するにつれて連続的に更新される。

#### 【0073】

- ・時刻 $t_2$ では、プロセッサ内のタイマから割込みINT1が受信される。

プロセッサは、ポインタ P t r T C I N T 1を、ポインタ・レジスタ 122に読み込み、その後で T H R E A D C O N T E X T Fを、このポインタにより特定されているメモリ位置から読み出す。キャッシュ・マネジャーは、このスレッド・コンテキストが低い優先順位のコード  $p = 0$  を含んでいることと、どのコンテキスト・レジスタ・セットも低い優先順位のコンテキストを含んでいないことと、コンテキスト・レジスタ・セット II、III、IVはフリーであること、を判定し、 T H R E A D C O N T E X T Fをコンテキスト・レジスタ・セット IIへ送る。（あるいは、コンテキスト・レジスタ・セットの特定の1つを低い優先順位のスレッド・コンテキストのために保留できる。これによって、コンテキスト・レジスタ・セットのいずれも低い優先順位のスレッド・コンテキストのいずれも含んでいない、という判定ステップが無くされる。）コンテキスト・レジスタ・セット IIはアクティブとなり、プロセッサはスレッド F の実行を開始する。

#### 【0074】

- 時刻  $t_3$  では、割込み E X T 1がバス 30 を介して受信される。プロセッサは、ポインタ P t r T C E X T 1をポインタ・レジスタ 122 に読み込み、その後で T H R E A D C O N T E X T Aをこのポインタにより特定されているメモリ位置から読み出す。キャッシュ・マネジャー 124 は、このスレッド・コンテキストが高い優先順位のコード  $p = 1$  を含んでいることと、コンテキスト・レジスタ・セット III と IV がフリーであること、を判定し、 T H R E A D C O N T E X T Aをコンテキスト・レジスタ・セット III へ送る。このレジスタ・セットはアクティブとなり、プロセッサはスレッド A の実行を開始する。

#### 【0075】

- 時刻  $t_4$  では、スレッド A の実行中に、プロセッサは命令 S W I T C H T O P 0 T Cを、現在コンテキスト・レジスタ・セット III にあるプログラム・カウンタ・レジスタ P C によって特定されているメモリ 20' 内のそれぞれの位置から読出す。（この例では、 T H R E A D C O N T E X T G のメモリ位置に対するアドレスがコンテキスト・レジスタ・セット III の P 0 レジスタ

に予め保持される。また、スレッドG内に含まれている、優先順位コード $p = 0$ によってキャッシュ・マネジャーが、このスレッド・コンテキストをコンテキスト・レジスタ・セットIIに読み込むべきであることを決定可能にされる。) この命令の実行においてはプロセッサは、コンテキスト・レジスタ・セットが1つだけ存在する場合におけるように、次のように処理する。

#### 【0076】

- ・ コンテキスト・レジスタ・セットII内のレジスタPTR\_TCに含まれている現在のスレッド・コンテキスト・アドレスにより特定されているメモリ位置にTHREAD\_CONTEXT\_Fを読み込み、コンテキスト・レジスタ・セットIII内のレジスタPTR\_TCに現在含まれているアドレスをレジスタPREV\_PTR\_TCに読み込む。

#### 【0077】

- ・ コンテキスト・レジスタ・セットIIIのP0レジスタからのTHREAD\_CONTEXT\_Gのアドレスを、ポインタ・レジスタ122に読み込む。

#### 【0078】

- ・ THREAD\_CONTEXT\_Gをコンテキスト・レジスタ・セットIIに読み込む。レジスタ・セットIIは今はアクティブとなり、プロセッサはスレッドGの実行を開始する。

#### 【0079】

- ・ 時刻t5では、割込みEXT\_3がバス30を介して受信される。プロセッサはポインタPtr\_TC\_EXT3をポインタ・レジスタ122に読み込み、その後でTHREAD\_CONTEXT\_Cをこのポインタにより特定されているメモリ位置から読出す。キャッシュ・マネジャー124は、このスレッド・コンテキストが高い優先順位のコード $p = 1$ を含んでいることと、コンテキスト・レジスタ・セットIVのみがフリーであることと、を判定し、THREAD\_CONTEXT\_Cをコンテキスト・レジスタ・セットIVへ送る。このレジスタ・セットはアクティブとなり、プロセッサはスレッドCの実行を開始する。

#### 【0080】

この時刻から、スレッド・コンテキストA、B、Cのそれぞれは、それぞれの

コンテキスト・レジスタ・セット III、I、IVに保持され、それらのスレッドのそれぞれは、それぞれのレジスタ・セットをアクティブにするだけで実行できる。スレッドD、E、F、G、Hのいずれかの実行にはそれぞれのスレッド・コンテキストをコンテキスト・レジスタ・セット IIにロードすることを要する。

### 【0081】

このように、高い優先順位のスレッドの数と少なくとも同数の専用の高い優先順位のコンテキスト・レジスタ・セット（たとえば、図3に表されている）で動作するように構成されている装置では、それらのレジスタ・セットにそれぞれのコンテキストがひとたびロードされると、それらとメモリ20の間でコンテキストを再び転送するいかなる必要も決してない。したがって、そのような転送に付随する遅延が避けられる。逆に、低い優先順位のスレッドに対してコンテキスト・レジスタ・セットが1つだけ設けられたとすると、それらのスレッドに対するコンテキストを、プロセッサがそれらのうちの異なる1つに変更するたびに单一のレジスタ・セットへ転送しなければならない。

### 【0082】

図4は、2つの優先レベルで動作する別のシステム構成におけるスレッド処理の例を表す。第1のレベルでは、図3の例におけるように、スレッド当たりより多くのレジスタ・セットがより低い優先順位のスレッドよりも高い優先順位のスレッドに対して使用できるようにされる。第1の優先レベルでは、複数の専用コンテキスト・レジスタ・セットを有する各スレッド優先順位コード $p$ に対してあふれ優先順位アルゴリズムが利用されるが、その場合にはそれらのレジスタ・セットの数がそれぞれの優先順位コード $p$ を持つスレッドの数より少ない。この例では、再び、

- ・ 高い優先順位 ( $p = 1$ ) のスレッドまたは低い優先順位 ( $p = 0$ ) のスレッドを指示するために单一のビットが利用される、しかし、今では
- ・ スレッドA、B、C、Dは高い優先順位（すなわち、 $p = 1$ ）とされ、割込みEXT 4に応答してプロセッサにより読出され、かつTHREAD

CONTEXT Dのメモリ位置を有する追加のポインタ P t r T C E X T 4がメモリ 20' に保存されており、

- ・スレッド E、F、G、H は低い優先順位（すなわち、 $p = 0$ ）とされる。

#### 【0083】

図 4 の例では、時刻  $t_0$  から時刻  $t_5$  まで、動作は図 3 について説明したものと同じである。しかし、その後では、第 2 のレベルの高い優先順位のスレッドの数より、高い優先順位のコンテキスト・レジスタ・セットが少ないので、第 2 のレベルの優先順位アルゴリズムが実行される。たとえば、高い優先順位のスレッドが相互に等しくない優先順位のものであるとすると、第 2 のレベルの優先順位アルゴリズムは、そのような相対的優先順位に基づくものでもよい。あるいは、それらがすべて等しい優先順位のものであるとすると、異なるアルゴリズムを利用してもよい。以下のアルゴリズムは、高い優先順位のスレッドによるプロセッサの過去の利用の履歴に基づいている。

#### 【0084】

・時刻  $t_6$  では、割込み EXT 4 がバス 30 を介して受け取られる。プロセッサはポインタ P t r T C E X T 4 を、ポインタ・レジスタ 122 に読み込み、その後で THREAD CONTEXT D を、このポインタにより特定されているメモリ位置から読み出す。キャッシュ・マネジャー 124 は、スレッド・コンテキストが高い優先順位のコード  $p = 1$  を含むことと、ただしコンテキスト・レジスタ・セットはどれもフリーでないことと、を判定する。キャッシュ・マネジャーは、各スレッドに対するキャッシュ・メモリを使用するための要求の履歴をトラッキングできる。好適な実施例では、これは、各スレッドに別々のカウンタを提供し、最初にそれらのカウンタの全てを零にリセットし、その後で、要求が行われるたびに、

- ・要求が行われたスレッドに関するカウンタのみを零にリセットし、
- ・他のスレッドに対するカウンタを増加する。

#### 【0085】

それらのカウンタに保持されている履歴から、最後の要求がスレッド B に対す

るものであったことをキャッシュ・マネジャーは判定する（割込みEXT 2が受信され、且つ、THREAD CONTEXT Bが、コンテキスト・レジスタ・セットIに読み込まれた時刻t1に、このスレッドBは生じたものである）。この履歴に基づいて、プロセッサはTHREAD CONTEXT Bをメモリ20'に読み込み、その後でTHREAD CONTEXT Dをコンテキスト・レジスタ・セットIに読み込む。このレジスタ・セットは今はアクティブとなり、プロセッサはスレッドDの実行を開始する。

#### 【0086】

多数のコンテキスト・レジスタ・セットを設けることによってプロセッサは、メモリ20をアクセスすることなくある数のスレッドを逐次処理する。各スレッド変更に必要な時間は、最も速いバスを介してメモリをアクセスするために必要とされるはるかに長い時間ではなくて、クロック・サイクルで測定される。

#### 【図面の簡単な説明】

##### 【図1】

本発明の一実施例に係る多重スレッド処理装置の動作を全体的に示す図である。

。

##### 【図2】

ポインタおよびスレッド・コンテキストの特定のセットがメモリに保存されている図1の多重スレッド処理装置の動作を示す図である。

##### 【図3】

優先順位付けされたスレッドでの図1の多重スレッド処理装置の動作を示す図である。

##### 【図4】

第1のレベルおよび第2のレベルの優先順位を有する図1の多重スレッド処理装置の動作を示す図である。

#### 【符号の説明】

10 プロセッサ

12 割込み処理器

14 コンテキスト・レジスタ・セット

16 命令復号器

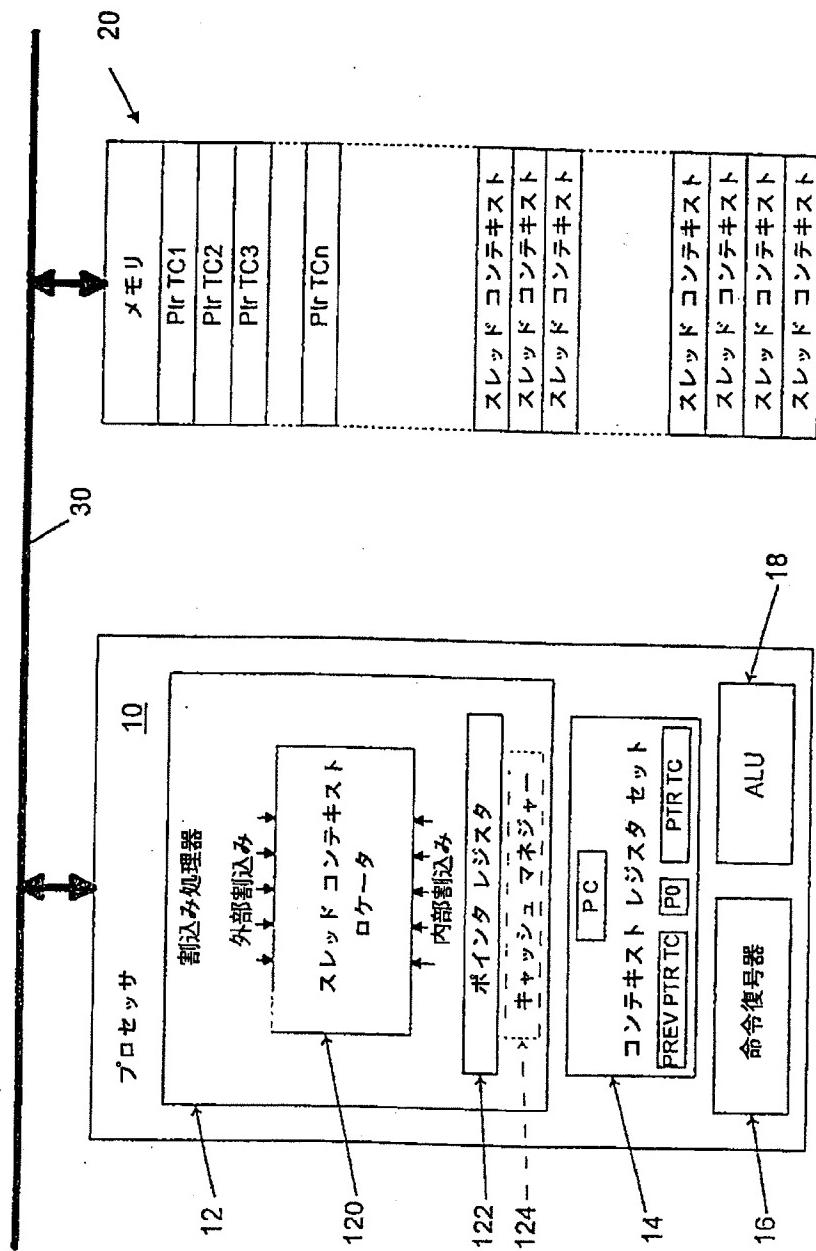
20 メモリ

120 スレッド・コンテキスト・ロレータ

122 ポインタ・レジスタ

124 キャッシュ・マネジャー

【図1】



【図2】

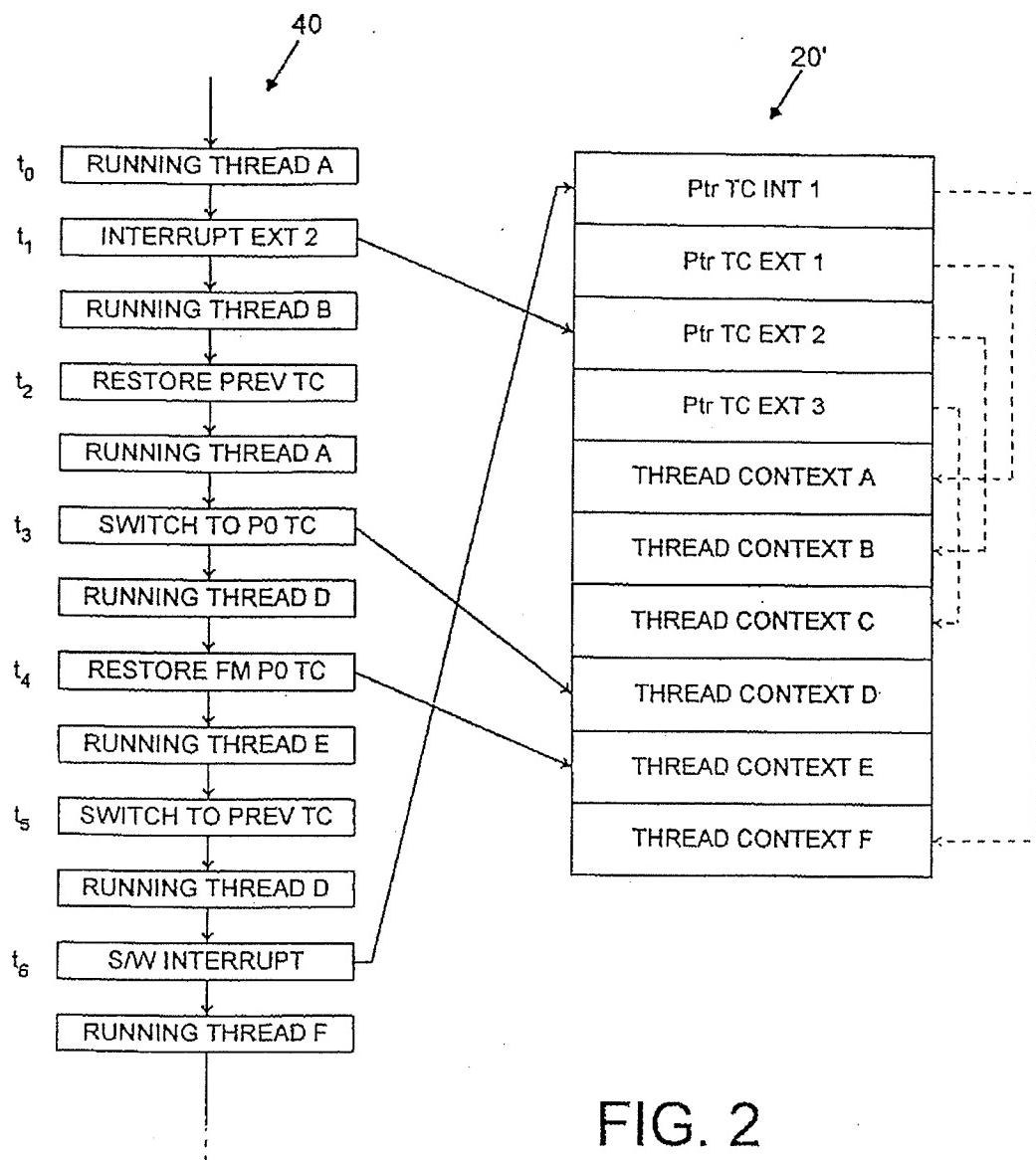


FIG. 2

【図3】

		I	II	III	IV
$t_0$					
$t_1$	EXT 2	B			
$t_2$	INT 1	B	F		
$t_3$	EXT 1	B	F	A	
$t_4$	SWITCH TO P0 TC	B	G	A	
$t_5$	EXT 3	B	G	A	C

FIG. 3

【図4】

		I	II	III	IV
$t_0$					
$t_1$	EXT 2	B			
$t_2$	INT 1	B	F		
$t_3$	EXT 1	B	F	A	
$t_4$	SWITCH TO P0 TC	B	G	A	
$t_5$	EXT 3	B	G	A	C
$t_6$	EXT 4	D	G	A	C

FIG. 4

## 【国際調査報告】

## INTERNATIONAL SEARCH REPORT

Int. Appl. No.  
PCT/EP 99/10170

A. CLASSIFICATION OF SUBJECT MATTER IPC 7 G06F9/46		
According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched: (classification system followed by classification symbols) IPC 7 G06F		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practical, search terms used) EPO-Internal		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 4 074 353 A (IZBICKI KENNETH J ET AL) 14 February 1978 (1978-02-14) column 10, line 56 -column 12, line 38 column 13, line 49 -column 16, line 48	1,2,4, 19,30 5-9,11, 12,20-25
X	US 5 659 749 A (MITCHELL BOB ET AL) 19 August 1997 (1997-08-19) column 2, line 16 - line 51 column 5, line 14 - line 26 column 6, line 18 -column 7, line 57	1,3,4,30
Y	US 5 696 957 A (HARA KAZUHIKO ET AL) 9 December 1997 (1997-12-09) column 6, line 8 - line 23	12 -/-
<input checked="" type="checkbox"/> Further documents are listed in the continuation of box C.		<input checked="" type="checkbox"/> Patent family members are listed in annex.
<p>* Special categories of cited documents:</p> <p>"A" document defining the general state of the art which is not considered to be of particular relevance</p> <p>"E" earlier document but published on or after the international filing date</p> <p>"L" document which may throw doubts on priority, claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)</p> <p>"O" document referring to an oral disclosure, use, exhibition or other means</p> <p>"P" document published prior to the international filing date but later than the priority date claimed</p> <p>"T" later document published after the international filing date or priority date and in conflict with the application but cited to understand the principle or theory underlying the invention</p> <p>"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone</p> <p>"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art</p> <p>"&amp;" document member of the same patent family</p>		
Date of the actual completion of the international search  23 June 2000	Date of mailing of the international search report  05/07/2000	
Name and mailing address of the ISA  European Patent Office, P.B. 5818 Patentlaan 2 NL - 2290 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl. Fax: (+31-70) 340-3016	Authorized officer  Bijn, K	

## INTERNATIONAL SEARCH REPORT

International Application No.  
PCT/EP 99/10170

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5 742 822 A (MOTOMURA MASATO) 21 April 1998 (1998-04-21) column 6, line 10 -column 10, line 5 -----	5-9,11, 22 10
A	US 5 349 680 A (FUKUOKA KATSUHITO) 20 September 1994 (1994-09-20) column 14, line 13 -column 20, line 21 -----	5,21
Y	US 4 410 939 A (KAWAKAMI KATSURA) 18 October 1983 (1983-10-18) column 3, line 20 - last line column 4, last line -column 6, line 37 -----	20,23-25

1

Form PCT/ISA/210 (continuation of record sheet) (July 1992)

## INTERNATIONAL SEARCH REPORT

Information on patent family members

Inten.	Final Application No
PCT/EP 99/10170	

Patent document cited in search report	Publication date	Patent family member(s)		Publication date
US 4074353 A	14-02-1978	BE	854924 A	16-09-1977
		CA	1084170 A	19-08-1980
		DE	2722099 A	08-12-1977
		FR	2353102 A	23-12-1977
		GB	1547312 A	06-06-1979
		HK	37280 A	18-07-1980
		JP	1237667 C	31-10-1984
		JP	52144243 A	01-12-1977
		JP	59011943 B	19-03-1984
US 5659749 A	19-08-1997	NONE		
US 5696957 A	09-12-1997	JP	5265753 A	15-10-1993
US 5742822 A	21-04-1998	JP	8171494 A	02-07-1996
		GB	2296354 A, B	26-06-1996
US 5349680 A	20-09-1994	JP	4172551 A	19-06-1992
		JP	2062324 C	24-06-1996
		JP	4182834 A	30-06-1992
		JP	7095276 B	11-10-1995
		KR	9512293 B	16-10-1995
US 4410939 A	18-10-1983	JP	56016248 A	17-02-1981

---

フロントページの続き

(81)指定国 EP(AT, BE, CH, CY,  
DE, DK, ES, FI, FR, GB, GR, IE, I  
T, LU, MC, NL, PT, SE), CN, JP, K  
R, SG

(71)出願人 Groenewoudseweg 1,  
5621 BA Eindhoven, Th  
e Netherlands

Fターム(参考) 5B098 AA10 BA06 BA12 CC02 GA05

GC03